2020-04-03

Data Curation Format Profile: MATLAB

Sciolla, Sam; Borda, Susan

Data Curation Format Profile: MATLAB

Research Data Services
University of Michigan Library

Contributor(s): Sam Sciolla (ssciolla@umich.edu); Susan Borda* (sborda@umich.edu)
*denotes corresponding creator

Core Details about MATLAB file formats	
File Extensions	.mat (matrix data) and .m (MATLAB script - simple text file)
MIME Type	application/x-matlab-data application/matlab-mat
Structure	MAT-file: Binary, short text header
Versions	Level 5 - Version 7.3=< [1] Level 4 Version 7.3 files are HDF5 datasets
Disciplines of use	Matlab is widely used in science and engineering disciplines
Transparency and affiliation	The MATLAB language, software and the MAT-File data format are developed by MathWorks, a commercial company founded in 1984. While licenses are expensive and the HDF5 version of MAT-File is not documented, online help material and documentation for the language is thorough. In addition, the website states that it allows MATLAB users access to a PDF documenting some versions of MAT-File; that file seems to be available even to those not logged in: https://www.mathworks.com/help/pdf_doc/matlab/matfile_format.pdf
Metadata standards	N/A?
Key questions	• How are MATLAB scripts and data files organized in the dataset? • How well does internal and external metadata explain how to understand and use the MATLAB files? • To what extent are all steps in the research process accounted for in the dataset?
Tools and procedures	• Gathering version and dependency information • Load, Save, Run • Exploring MAT-File structures • Open Source options - GNU Octave and Python
Date Created	July 2, 2018
Date updated and summary of changes made	Aug 2, 2019 - images added for clarity and more information about open source resources. S.B. April 2, 2020 - added Python code snippets and info about Search Path. S.B.

[1] MAT-File Level 5 File Format - https://www.loc.gov/preservation/digital/formats/fdd/fdd000440.shtml

Table of Contents

Description of the format(s)

MATLAB is both a programming language and a software environment developed by MathWorks for use by scientists and engineers. The technology enables complex mathematical operations, data analysis, and visualization through built-in graphics functionality. Short for "matrix laboratory," MATLAB stores all data in variables as matrices, or two-dimensional arrays, according to the online documentation (MathWorks 2018 - Getting Started with MATLAB).

The Library of Congress lists common areas of use as "chemical engineering, bio-engineering, signal processing (including for images), medical image analysis, quantitative finance, pattern recognition," but our interaction with researchers studying geology and atmospheric and space sciences suggests that MATLAB's capabilities appeal to perhaps a more diverse group of disciplines and research areas. (Library of Congress 2017).

MATLAB allows for the creation of two different file formats: .m and .mat. MATLAB users save scripts using the .m format, which can be opened using a general purpose text editor. Data created or saved within MATLAB can be stored in MAT-Files (which have the extension .mat), a proprietary binary format that has seen a handful of iterations. Documentation is available to current users that provides technical specifications by level, with Level 5 being the most recent (MathWorks 2018 - MAT-File documentation). However, the MATLAB help documentation defines format iterations by version; the save function in MATLAB automatically creates MAT-Files in Version 7, but there is also a Version 7.3, based on the HDF5 format (MathWorks 2018 - MAT-File Versions).

The Level 5 file specifications describe a general file structure of a brief header (which can be viewed with a text editor as in **Figure 1**) and then a series of data elements, each of which include a tag specifying the data type contained in the element and the size of the element in bytes. Data types can be numerical (integer or floating-point numbers of different sizes, signed or unsigned), Unicode text, compressed data, or a MATLAB array.

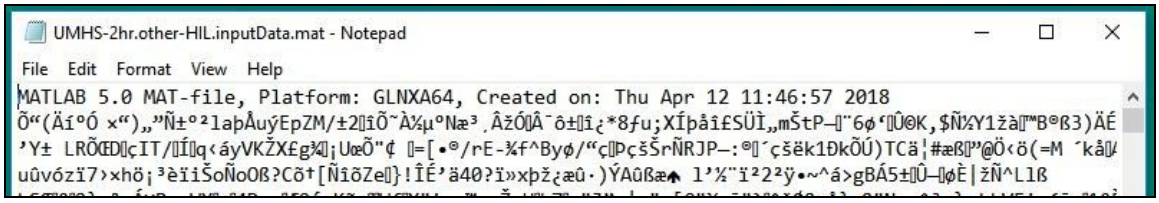


Figure 1: The text header of a MAT data file in Notepad on a PC

The last of these can contain a number of special MATLAB data types including structures (in which named fields are mapped to values), arrays (of homogeneous and heterogeneous values), and objects (as in object-oriented programming). MATLAB arrays can also be nested within other MATLAB arrays. The online help documentation contains detailed explanations of and instructions on using each of these in the "Data Types" section (MathWorks 2018).

Example of Datasets in MATLAB

https://deepblue.lib.umich.edu/data/concern/data_sets/j67314445?locale=en
https://deepblue.lib.umich.edu/data/concern/data_sets/qr46r1420?locale=en

Key questions to ask when reviewing MATLAB datasets

Datasets containing MATLAB files can differ greatly in appearance, as researchers use the powerful computing tool to perform various mathematical, analytic, and visualization-related tasks. Datasets can include numerous interdependent scripts and incorporate data from MAT-Files with complex, heterogeneous structures.

Rather than creating a list of characteristics that "well-formed" MATLAB files need to have, we have created a list of questions that we hope will help data curators determine the clarity and usefulness of a MATLAB dataset to a discipline and direct conversation with researchers on ways it — and the .m and .mat files within it — might be improved.

For each overarching question, there will be a statement of what your goal will likely be as a data curator and reviewer of the dataset; a series of sub-questions to direct your investigation; and some background information, gleaned from our own research and conversations with researchers.

It will be easier to answer these questions once you have reviewed specific characteristics of the MATLAB files in question. For .m scripts, you will want to examine file relationships, important functions, dependencies, and comments. For .mat files, you will want to ascertain the data types within them and gather information from variable names. For an overview of the procedures and tools we recommend using to accomplish these tasks, read the next section, **Instructions for reviewing MATLAB files**.

1. How are MATLAB scripts and data files organized in the dataset?	Goal	To assess how well directories, file-naming conventions, and scripts/functions are used to make the relationships between dataset components transparent
	Questions to answer	• How are directories and subdirectories used to compartmentalize input, output, and scripts accomplishing different workflow steps? • Do file and directory names help users in identifying the different parts of the workflow? • Do scripts appear to rely on each other, and if so, is there a hierarchy, with one or two files referencing others?
	Background	Programmers in high-level languages like MATLAB have a myriad of ways to structure a dataset or codebase. Functions defined in one script can be used in another, an arrangement that is a core programming concept called modularization. The authors of the article "Best Practices for Scientific Computing" recommend modularization in order to make programs more human-readable, less repetitive, and easier to de-bug (Wilson et al. 2014). In addition, data input or output files can be pointed to and accessed from separate directories along the same file path. These practices can be seen in the complex datasets deposited by two researchers we spoke with (Researcher A 2018; Researcher B 2018). Researchers can further improve a clear dataset structure by giving files and directories descriptive names that follow a consistent style, are meaningful to the content they contain and denote how files or directories are related. To help programmers and code readers distinguish between entities, Wilson et al. suggest that "scientists ***make names consistent, distinctive, and meaningful***" (Wilson et al. 2014). To achieve consistency at the file level, names might use the same character (or delimiter) to separate words and the same part of speech (probably a noun or verb) to articulate what the code is accomplishing. Ideally, a README document or introductory comments in a central script would explicitly describe the role of each file in a dataset and the relationships between them.

2. How well does the internal and external metadata explain how to use and understand the MATLAB files?	Goal	To assess whether the information provided through comments in scripts, the names of variables in data files, and other content in README files would enable a likely future user to use and understand the dataset
	Questions to answer	• Have researchers indicated in comments or in a README file the version of the MATLAB language the programs are written in, the versions of any included MAT-Files, and any specialized toolboxes used? • Are there instructions on how to run scripts, including explanations of input parameters and accepted values? • Do the researchers indicate how long important scripts might take to run, what hardware has run them successfully, and whether progress is saved in the case of an interruption? • If a MAT-File is included and contains elements of MATLAB's structure data type, are variables named clearly and described elsewhere (e.g. definitions and units are provided in a README)?
	Background	When documenting how MATLAB code works and how to use it, researchers have the choice of embedding information in scripts as comments and/or creating an external README file. While Wilson et al. recommend that documentation be embedded in code to increase its accessibility and likelihood of being updated, the researchers we spoke with often chose to create a README in addition to commenting (Wilson et al. 2014). Rather than recommend a particular approach, we encourage consistent and detailed documentation that can be easily found and referred to by data re-users. A major imperative of code documentation is to specify how to run the core scripts and initiate processes made possible by the code. Because MATLAB scripts that define a function can accept input or a parameter (see the Instructions section below), it is often critical for the accessibility of a program to document accepted values and the situations where each might be used. In addition to instructions on executing scripts, users may also need further software and hardware to assist with

		troubleshooting and setting expectations. The researchers we spoke with were optimistic about MATLAB's backwards compatibility and felt that they primarily used the language's core functionality (Researcher C 2018; Researcher B 2018; Researcher A 2018). However, given the profusion of special toolboxes available and the pace at which programming languages have changed over the last couple decades, it seems useful to detail version numbers and dependence on toolboxes. MATLAB includes a simple function that lists a script's version and requirements (see the Instructions section below). In many circumstances, MATLAB scripts are written to perform computationally expensive tasks and can take hours or even days to complete. A couple of the researchers we have interacted with have included in documentation details about expected run times and details about the hardware the scripts have been tested on (e.g. operating system and memory, which can affect memory)(Researcher B 2018; Researcher A 2018). These details can help users plan ahead and locate other computing resources if necessary. Lastly, in cases in which researchers use MAT-File (.mat) data files, significant details may be contained within the file itself. MATLAB software allows for exploration of these files, and field names can be present when MATLAB's structure data type is used (MATLAB - struct). This information may illuminate some aspects of the scripts and research, and it may also suggest the need for further documentation elsewhere, such as a data dictionary.

3. To what extent are all steps in the research process accounted for in the dataset?	Goal	To assess whether the entire research process — from raw data or configuration files to results and/or visualization — is captured by the files and documentation in the data set.
	Questions to answer	• Are steps in data processing and analysis represented one-to-one by scripts or functions (e.g. reading and plotting routines)?

		• Does the dataset include initial files (raw data or configuration files for simulations) and document how those were acquired or determined? • In accompanying documentation, do the researchers identify clearly how any outside resources were used during the research process, including other datasets, supercomputers, and/or licensed code or programs?
	Background	As a tool useful across domains for advanced mathematical analysis, MATLAB can play many different roles in the research process. Data can begin and end their existence within MATLAB or be imported from other files — .csv files, netCDF files, binary files generated by supercomputers are just a few examples — before undergoing any number of layers of processing, transformation, and presentation. Because of that flexibility, MATLAB datasets benefit especially from careful organization of components and documentation of provenance. MATLAB programs can be split up into numerous smaller scripts. In many cases, these code snippets and functions can represent particular steps in the data manipulation process (an example of modularization, discussed in Question 1). For instance, one researcher we spoke with discussed "reading routines versus plotting routines," differentiating between a program that transforms raw data into a usable form and another program that plots and/or analyzes that data for scientific purposes (Researcher B 2018). Another researcher created separate scripts (or functions) for each plot that appears in a paper and others that implement special data analysis algorithms (Researcher C 2018). Equally important to capturing provenance is including and documenting data files, configuration or job files, and other resources used throughout the research. Datasets reviewed by data curators may or may not include data in its most raw, 'just collected' form, as was the case with one researcher we spoke with who was part of a patent application (Researcher A 2018). Researchers using supercomputing resources may not include the files that contained the parameters used to run the remote job; one researcher we spoke with did not choose to deposit such a

		file (Researcher C2018). When data collected or code written by others are employed, researchers may or may not provide citations or licenses; a researcher we interviewed included licenses for others' code he used but did not formally cite a few sample datasets included in the deposit (Researcher B 2018). While transparency is desirable (and in many cases feasible through documentation), the importance to data re-users of full reproducibility may be less clear. Discussions with researchers revealed that different dataset components were perceived to be more likely to be reused, with one seeing more value in the data, another in the programs or algorithms employed, and a third in both (Researcher A 2018; Researcher B 2018; Researcher C 2018). Discussions with researchers can help to identify anticipated uses of the data, as well as possible barriers researchers face in representing all steps in a provenance chain.

Instructions for reviewing MATLAB files

As a major component of reviewing any dataset is to ensure files are functional, reviewing MATLAB datasets requires some knowledge of how to run scripts in MATLAB and view data files. This section covers the basics of file interaction in MATLAB and how to identify version or dependency issues you or other users may encounter.

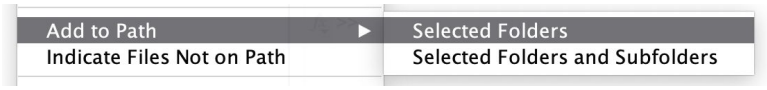
Release version used in this document: MATLAB R2017a (on Windows)

Sample dataset used in the examples below

1) *Running scripts*

To run a MATLAB script, you must know the location of the script you wish to run and navigate to the correct directory within the MATLAB interface. After opening the MATLAB application, use either the Current Folder panel on the left or the file path toolbar directly above it to find the desired directory.

** Having files (scripts and input files) in the "Search Path" is very important in MATLAB, this can be done via contextual menu (right click) "Add to Path" > "Selected Folders"



or via the Environment menu “Set Path”.

The most basic scripts can be run easily by right-clicking on a MATLAB script in the Current Folder panel and then selecting “Run.” Scripts can also be run using the Command Window (at the center or center bottom of the interface). Type the name of the file without the file extension and hit Enter. For instance, to run, “file_name.m”, you would enter the following (**Figure 2**):

```
>> file_name
```

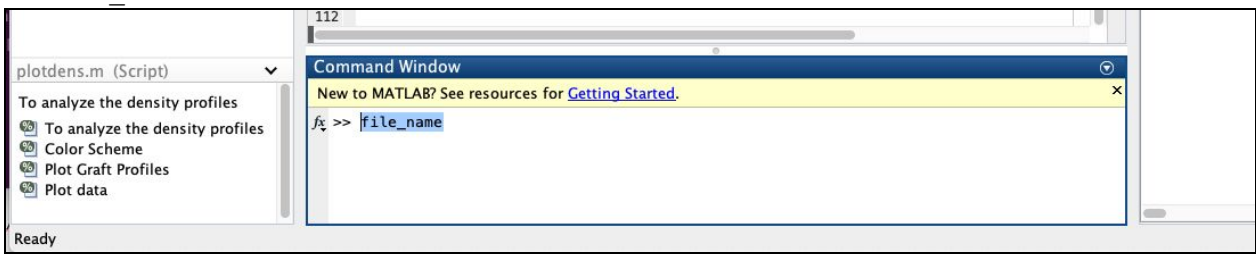


Figure 2: Running a *.m script from the Command Window

Problems can arise when a MATLAB script is associated with a function defined within the script. Functions in MATLAB operate similar to how they do in other programming languages, often accepting input parameters and returning outputs. However, unlike in other languages, a function in MATLAB that shares the name of the script it is defined in can be called directly from the Command Window.

Functionally this results in MATLAB scripts sometimes requiring inputs, which can cause errors when trying to run using the right-clicking method. To run the script successfully, you will have to review the file, comments therein, and any documentation for the dataset to determine what valid inputs for the file might be. In our experience working with MATLAB datasets, we have often seen the names of data files serving as inputs to functions.

Once you have found a valid input that you would like to use, type the name of the script in the Command Window as described above and then follow it immediately with the name of the desired input enclosed between single quotes and parentheses. The command will look something like the following:

```
>> file_name('input')
```

Scripts can set into motion any number of processes, including the generation of data files or visualizations. You may also have to be patient, as we have encountered MATLAB scripts that can take hours to run and have heard of others taking days.

2) Loading, viewing, and saving data

MAT-Files — MATLAB's proprietary binary data format — are fairly straightforward to work with in the MATLAB interface, though some knowledge of the language's data types can be useful. The "Data Types" section of MATLAB's online documentation provides a detailed overview of the various data types supported by the software.

To load a MAT-file in MATLAB, navigate to the desired directory and then simply double-click on the file in the Current Folder panel. Alternatively, you can use the load function in the Command Window, which would like the following, where "data_file_name.mat" is the file you want to load:

```
>> load('data_file_name.mat')
```

After executing the command, you will see a new item appear in the list in the Workspace panel on the right side of the interface. By clicking on the item, you will see the top level of the data file, which could be any data type. However, many complex data files will use the structure or cell array data types to organize the data and nest other instances of those types within them, as shown in the example in **Figure 3**. The data file can be further explored by clicking on cells with text in blue italics, which indicates a nested matrix of the dimensions and type indicated.

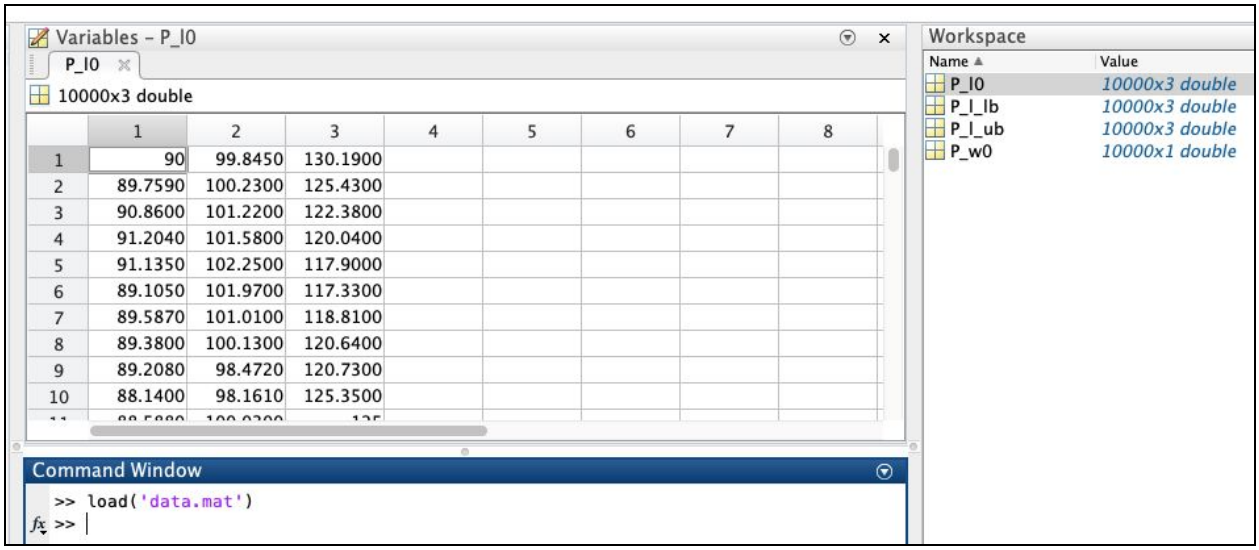


Figure 3: Example of a MAT-file using a structure data type, as viewed within the MATLAB interface

In some cases, scripts provided by researchers may generate data itself or convert data stored in one format to MATLAB data types so it can be analyzed and manipulated. Because some scripts can take a significant amount of time to run, it may be valuable to save and store data for future reference (if, of course, the researcher has not already included those files in the dataset).

After running such a script, the data produced should be saved in variables and visible in the Workspace panel on the right. To save that data to a MAT-File, use the Current Folder panel or the file path toolbar to navigate to the directory you want to save the file in, and then simply enter the following in the Command Window, where "file_name.mat" is the desired file name:

```
>> save("file_name.mat")
```

For more details on using the save function, such as how to save only a few variables currently in the workspace, see this page in MATLAB's online documentation.

2) *Gathering technical metadata: creation, version and dependency information*

When completing a full review of a MATLAB dataset, you will want to make note of the versions of any MAT-Files used, the version of the MATLAB language scripts were written in, and any specialized libraries or functions employed in those scripts. Recording these details can highlight possible accessibility issues to current and future users of the dataset (see Question 2 in Key questions to ask when reviewing MATLAB datasets).

MAT-files (.mat)

Though MAT data files are binary, the 128-byte header can be read using a text editor (e.g. NotePad, Text Edit, TextWrangler, Notepad++, etc.--note that files may be large enough to require text viewers with the capacity to handle large files, e.g., Universal Viewer). An example is shown in **Figure 1**. The header, includes three main details:

- the name of the file format, which is specified by level, not version
- the platform the file was created on (a code referring to the operating system and whether it is a 32- or 64-bit processor) → GLNXA64 (Linux - 64 bit), WIN64 (Windows - 64 bit), MACI64 (Mac Intel - 64 bit)
- and a timestamp indicating when the file was created

MATLAB scripts (.m)

The ability to successfully run the code contained in MATLAB scripts can depend on several factors, including the software version and MATLAB toolboxes (which enable specialized functions) used by the program.

MATLAB has a few built-in tools for viewing and capturing dependency information about individual scripts or entire directories, two of which are described briefly below and explained in detail on the "Identify Program Dependencies" MATLAB documentation page.

The requiredFilesAndProducts method allows you to view the version of MATLAB and any toolboxes used by a single script or an entire directory. To use the method, navigate using the Current Folder pane on the left of the MATLAB interface to the directory containing the file or files you want to analyze. If you want to analyze a directory, navigate to the directory above it in the file path, where you can see the directory listed. Then type the following in the Command Window, where "file_name.m" is the name of the file or directory:

```
>> [fList, pList] = matlab.codetools.requiredFilesAndProducts('file_name.m')
```

The output (**Figure 4**) will have two components:

- an array (or list) containing the files that were analyzed
- a structure (which will look like standard tabular data) listing the MATLAB products used, their version and product numbers, and a measure of how certain the method is that the resource is needed by these files

Figure 4: Output from the "requiredFilesAndProducts" method

Unfortunately, the documentation page states that this method only detects toolboxes which are already installed in your instance of MATLAB. This makes the tool valuable for the file creators or those who can already run the files but less helpful for those who may want to re-use the scripts. It may be helpful to ask researchers affiliated with a dataset to run this method on their local computing environment and share the results, which could then be added to their dataset.

The second option, running a Dependency Report, produces more comprehensive results which can also be useful in identifying relationships between scripts (see Question 1 in Key questions to ask when reviewing MATLAB datasets). According to the documentation page, this kind of report should flag missing files, which may help to identify missing toolboxes (MathWorks 2018).

To run a Dependency Report — which can only be run on a directory — navigate to the desired directory in the Current Folder panel; click on the down arrow on the right of the blue bar at the top of the panel; hover over Reports; and click on "Dependency Report."

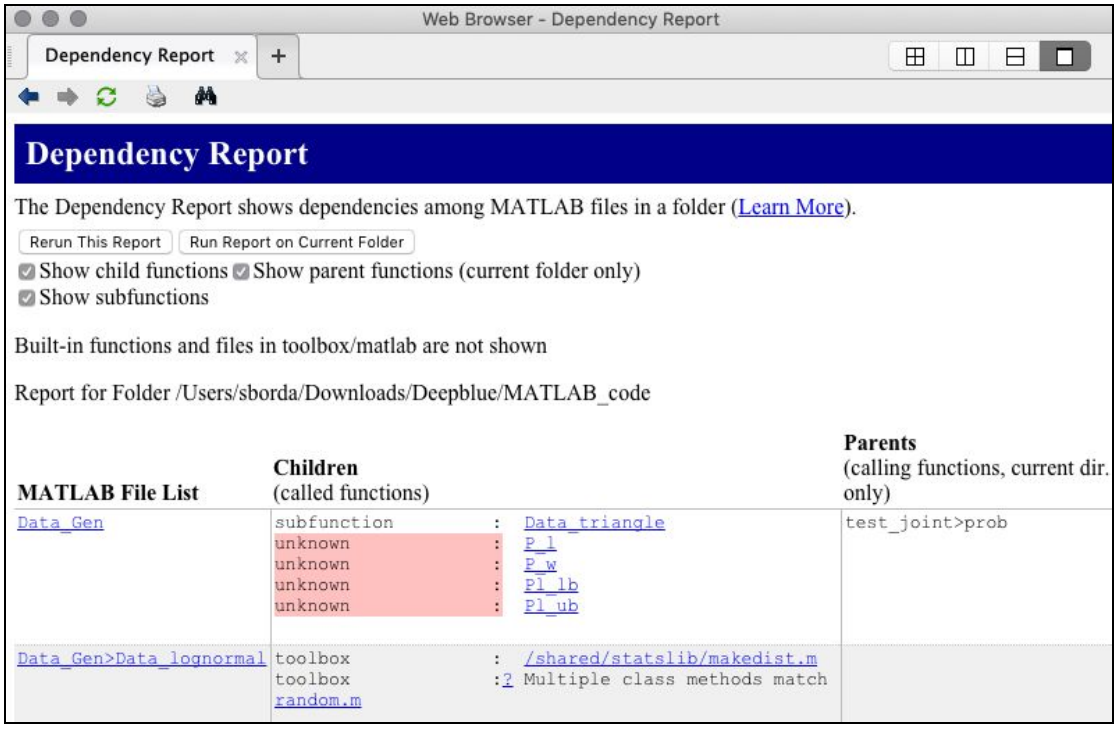


Figure 5: Example of a Dependency Report run on a directory containing numerous MATLAB scripts

By default, the pop-up window will look something like **Figure 5**, with three columns (only tow are shown) listing the following:

- the files analyzed in the report
- the "called functions," including a description of and the path to its source (e.g. "current dir" or "toolbox"
- the "calling functions," which indicate what higher-level files rely on the MATLAB file in question

The Dependency Report does not appear to search directories within the selected directory, so you may need to run the report multiple times on other nested directories.

Appendix A: Open Source alternatives (GNU Octave and Python) for working with MATLAB files

Octave is an open source alternative to MATLAB and uses the same using the same syntax as MATLAB to run *.m scripts and open *.mat data files
https://www.gnu.org/software/octave/

```
>> load('data_file_name.mat')
>> file_name('input')
```

Python can be used to view *.mat files using the libraries using the "scipy" library: https://scipy-cookbook.readthedocs.io/items/Reading_mat_files.html

```
import scipy.io
mat = scipy.io.loadmat('somefile.mat')
mat.items()
```

For MATLAB version 7.3 files use "numpy" instead of "scipy" to open them:

```
import numpy as np
import h5py
f = h5py.File('somefile.mat','r')
data = f.get('data/variable1')
data = np.array(data) # For converting to a NumPy array
```

MATLAB also has its own Python libraries for interacting with MATLAB and the MATLAB API:
https://www.mathworks.com/solutions/matlab-and-python.html

Bibliography

Interviews

Researcher C. Interview. University of Michigan, Climate and Space Sciences and Engineering. Held on June 1, 2018.

Researcher B. 2018. Interview. University of Michigan, Climate and Space Sciences and Engineering. Held on June 18, 2018.

Researcher A. 2018. Interview. University of Michigan, Climate and Space Sciences and Engineering. Held on June 20, 2018.

Web Resources

About the company
https://www.mathworks.com/company.html?s_tid=hp_ff_a_company

Matrices and Arrays
https://www.mathworks.com/help/matlab/learn_matlab/matrices-and-arrays.html

Data Types
https://www.mathworks.com/help/matlab/data-types_data-types.html

Identify Program Dependencies

https://www.mathworks.com/help/matlab/matlab_prog/identify-dependencies.html#responsive_offcanvas

MAT-File documentation
https://www.mathworks.com/help/pdf_doc/matlab/matfile_format.pdf

MAT-File versions (as discussed in online documentation)
https://www.mathworks.com/help/matlab/import_export/mat-file-versions.html

PRONOM page for MAT-File
http://www.nationalarchives.gov.uk/PRONOM/Format/proFormatSearch.aspx?status=detailReport&id=1606&strPageToDisplay=signatures

LOC page for MAT-File
https://www.loc.gov/preservation/digital/formats/fdd/fdd000440.shtml